

ThePoorEngineer

WORKING HARD TO BE LAZY

MENU

BUILDING THE CIRCUIT AND THE HARDWARE

Posted on February 4, 2018

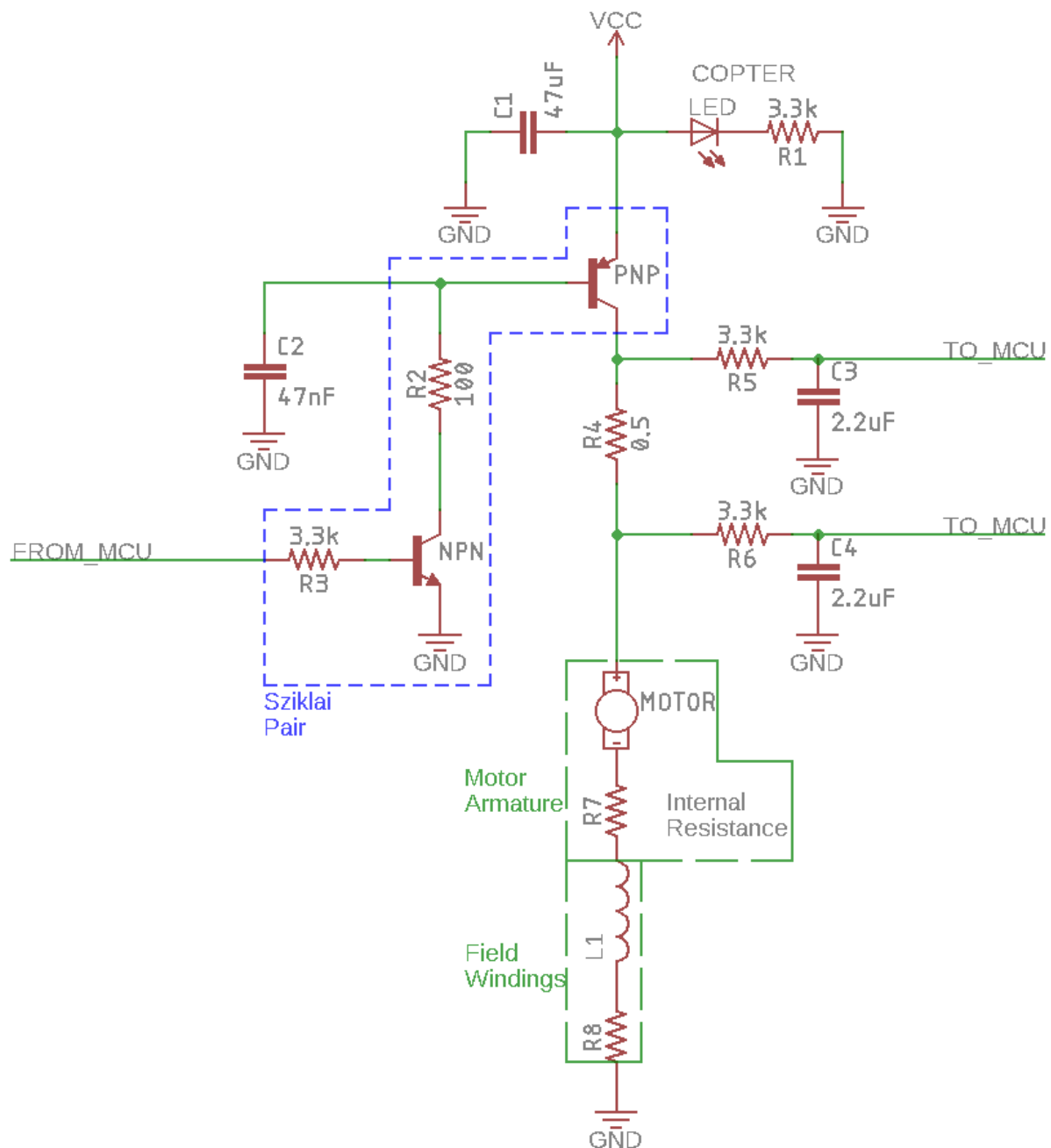
Table of Contents

1. Foreword and Introduction
2. Hardware and Software Requirements
 - Hardware List
 - Installing Pycharm with Anaconda
3. **Building the Circuit and the Hardware**
 - **Darlington/Sziklai Pair Transistors**
 - **Understanding Capacitors in the Context of Filtering**
 - **Putting Everything Together**
4. Plotting Serial Data from Arduino in Real Time with Python
 - Arduino Code
 - Python Code
 - Running the Programs
 - Running Multiple Plots at Once
5. Creating a Graphic User Interface (GUI) with Python
 - Combining Matplotlib with Tkinter
 - Understanding object placement in Tkinter
6. Motor Speed Control
 - Mathematical Model
 - Equation of Motion
 - Calibration
 - Proportional Control
7. Copter Angle Control (Relative)
 - Mathematical Model
 - Equation of Motion
 - Calibration
 - Proportional (P) Control
 - Proportional and Derivative (PD) Control
 - Proportional, Integral and Derivative (PID) Control
8. Copter Angle Control (Absolute)
 - Measuring the Angle relative to the ground
 - Understanding the Kalman Filter
 - Kalman Filter in Action with an Accelerometer and Gyrometer
 - Copter Absolute Angle Control

9. Conclusion

The Schematics for the circuit is provided right below this. If you already basic understanding of circuits, you can build the circuit from the schematics and skip this chapter. Otherwise, you should take some time to read this post which explains about the functions of common components in the circuit.

Here's the schematic for the circuit that we will be building. It will look really complicated at first for those who are not used to seeing them, but once you get a hang of it (hopefully after reading this post) it is actually not that difficult. I wrote the explanation for how this circuit works in the **last section** of this post so you can skip to there if you already know about transistor switching and capacitor filtering.



CIRCUIT SCHEMATIC

Darlington/Sziklai Pair Transistors

A transistor is one of the basic component in an electrical circuit and it's main purpose is to perform electrical switching. It is similar to a physical switch in that it allows you open/close the circuit but a transistor uses an electrical signal as an input to do this switching. There is already a [really good article from sparkfun](#) which explains the basics of a transistor so I will not talk about it here. Instead, I would like to talk about something that is more interesting, the Darlington/Sziklai Pair Transistors. If you take a look at the schematic diagram above once again, you will see that there are 2 transistors that are joined together. Why do you need a multi-stage transistor? The answer to this is the current amplification factor of the transistors.

If you look at the data sheet for transistors (here's [one](#)), you will see find a parameter labeled as h_{fe} (current amplification factor) such as the one shown below.

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
h_{FE}^*	DC Current Gain	$I_C = 0.1 \text{ mA}$ $V_{CE} = 1 \text{ V}$ $I_C = 1 \text{ mA}$ $V_{CE} = 1 \text{ V}$ $I_C = 10 \text{ mA}$ $V_{CE} = 1 \text{ V}$ $I_C = 50 \text{ mA}$ $V_{CE} = 1 \text{ V}$ $I_C = 100 \text{ mA}$ $V_{CE} = 1 \text{ V}$	60 80 100 60 30		300	

h_{fe} PARAMETER TAKEN FROM A SMALL SIGNAL **TRANSISTOR DATASHEET**

Just as the name suggests, this parameter tells you how much the transistor is able to amplify the current which you use to control the switching of the transistor. I_C is the current of the circuit that you want to control (current at the Collector) and I_B is the current of your control signal (current at the Base). The equation governing I_C and I_B is as follows:

$$I_C = h_{fe} * I_B \quad (1)$$

In the data sheet extract above, the supplier only gave values for up to $I_C = 100\text{mA}$ but lets say the max DC Current Gain value of 300 applies to all I_C values. Then, if you use an Arduino with a max current output of 0.020A, you can essentially control a circuit of at most $0.020\text{A} * 300 = 6\text{A}$. This means that it will be impossible for you to switch on a motor that requires 7A. In addition to this, you should never use the maximum values as a gauge for any designs for safety. Anyway, this is not really a good example. This type of small transistor is not designed to control motors that require huge amount of currents. In fact, the maximum value that your circuit should handle is the maximum value that is stated in the data sheet, which is in this case is 100mA.

Let us now take a look at **another transistor** which is designed for switching motors.

Characteristic	Symbol	Min	Typ	Max	Unit
DC Current Gain ($V_{CE} = 1.0 \text{ Vdc}$, $I_C = 2.0 \text{ Adc}$) ($V_{CE} = 1.0 \text{ Vdc}$, $I_C = 4.0 \text{ Adc}$)	h_{FE}	60 40	- -	- -	-

h_{fe} PARAMETER TAKEN FROM A POWER SWITCHING **TRANSISTOR DATASHEET**

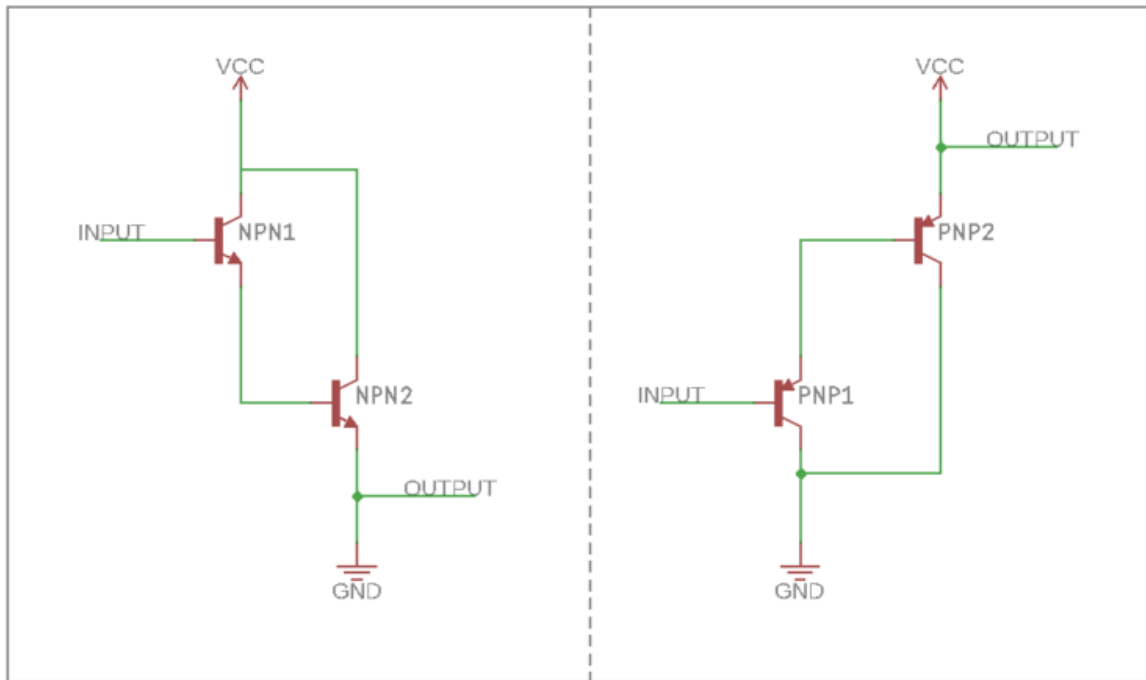
You can see that there are values given up to 4 Amperes in this datasheet for the transistor. The minimum h_{fe} for $I_C = 4.0\text{A}$ is 40 so using equation (1) from above, we know that $I_B = 4.0/40 = 0.1$. Since $h_{fe} = 40$ is a minimum value, if we give $I_B = 0.1$, it is guaranteed that we will definitely be able to

switch a 4A circuit. Doing the same analysis again using the 2A values, we know that we will need a current of at least 0.033A to switch a 2A circuit.

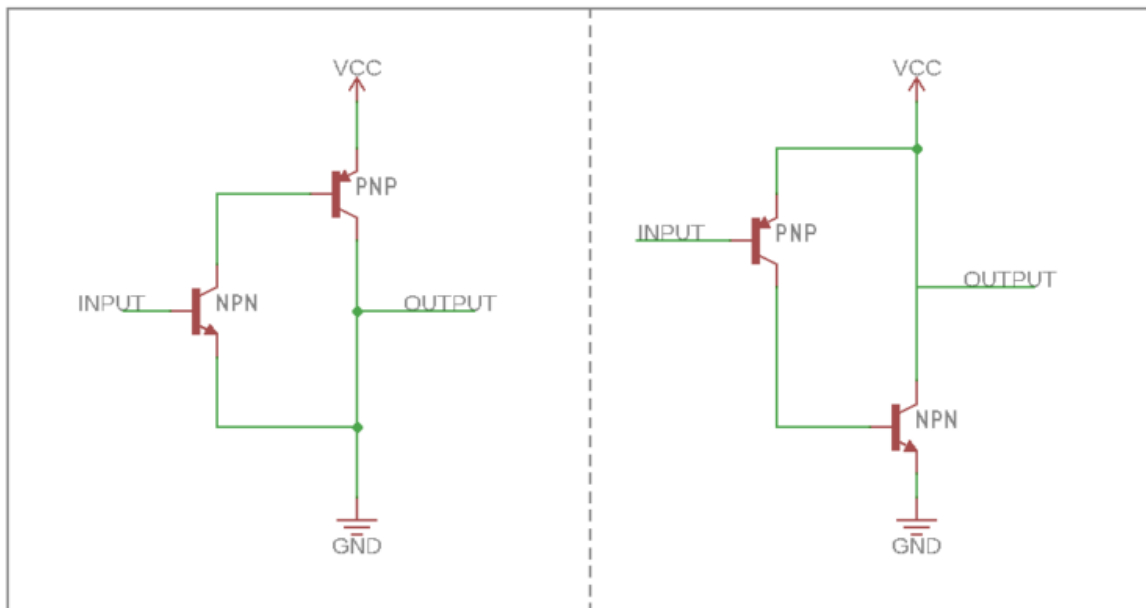
Now, we have a problem. Our micro-controller, the Arduino, is not able to supply 0.033A from its pins (maximum of 0.040A but usage of 0.020A is recommended). The motor that we are going to use is going to be taking about 3A when it is running at 5V so we know that the Arduino will NOT be able to switch OFF the motor (not ON because the transistor is a PNP type transistor) if we only use this transistor.

The solution to this is simple. Just add another transistor in the circuit to amplify the Arduino pin current first, then feed it into the power transistor. There are some consequences for doing this but it generally does not affect simple circuits like the one that we will be using. When you use 2 same type transistors in conjunction, it is called a **Darlington Pair** (PNP-PNP or NPN-NPN), and when you use 2 different types of transistor, it is called a **Sziklai pair** (PNP-NPN or NPN-PNP).

Darlington Pair



Sziklai Pair



DARLINGTON/SZIKLAI PAIR

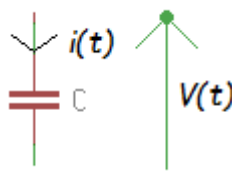
I have marked out the transistor section in our circuit schematic and we are actually using the Sziklai pair. Why the Sziklai and not the Darlington you ask? Not much reason here for our simple circuit. You can use a Darlington pair to do the same job with little, if not no difference in the final performance.

The 2 examples that I provided above are actually the transistors that we are going to use. As you can see from the h_{fe} variable, the small signal transistor can only safely control a circuit of 100mA so it will not be able to switch a motor circuit that requires approximately 3A. On the other hand, the power transistor requires an input current of approximately 0.033A to control a 2A circuit but our Micro-controller is only

able to supply up to 0.020A. The solution to this is to use both in conjunction in the circuit. We will first use the small signal transistor (2N3904) to amplify the current from our Arduino, and then use the amplified signal to switch the power transistor (D45H11) to control our motor.

Understanding Capacitors in the Context of Filtering

This topic may seem intuitive for those who have studied signal processing but for those of you from other backgrounds, I am guessing that you often have problems determining what value of capacitor to use for your circuit. I am writing this in hope this will help you understand the function of a capacitor, and also the impact it has on a circuit. I have written a python code to help you visualize the properties of a capacitor so if you do not have a python IDE yet, you can follow the instructions from my previous post for [installing Pycharm](#).



CAPACITOR

The diagram for a capacitor is shown above, and the mathematical model for the capacitor is given by:

$$i(t) = C \frac{dV(t)}{dt}$$

Fourier Transform tells us that any signal can be decomposed into sinusoids. If you haven't heard of the Fourier Transform, [this website](#) gives a simple and intuitive explanation to the very complex mathematical expressions of the Fourier Transform. The point here is that if all signals can be represented by sinusoids, then we can simply analyze any system by determining its response to sinusoids. In addition to this, Euler made our lives much simpler by providing us with the Euler's Formula:

$$e^{j\theta} = \cos(\theta) + j \sin(\theta)$$

, where j represents $\sqrt{-1}$

With a little manipulation, we find that it is possible to represent cosines and sines in exponential form using the Euler's Formula.

$$\cos(\theta) = \frac{1}{2} e^{j\theta} + \frac{1}{2} e^{-j\theta}$$

$$\sin(\theta) = \frac{1}{2} e^{j\theta} - \frac{1}{2} e^{-j\theta}$$

So what is the point in doing all these? First, by Fourier Transform, any signal can be represented by sinusoids. Then, via the Euler's Formula, we know that sinusoids can be represented by exponentials. Thus we can now analyze any system using these exponentials. The good thing about exponentials is that it is much simpler to deal with in differential equations so we have to thank Fourier and Euler for making our lives much simpler.

If we let $\theta = (2\pi f)t$, then

$$\cos((2\pi f)t) = \frac{1}{2}e^{j(2\pi f)t} + \frac{1}{2}e^{-j(2\pi f)t}$$

$$\sin((2\pi f)t) = \frac{1}{2}e^{j(2\pi f)t} - \frac{1}{2}e^{-j(2\pi f)t}$$

we can now represent signals of frequency f with the above equations. We can view the above equations of sine and cosine as having both positive and negative frequencies. This is because we are now working in both the real and imaginary domain. If you expand out the exponential terms again, you will see that the imaginary terms will cancel away leaving you with just the real value.

To put it generally, a sinusoid is made up of a positive frequency term ($e^{j(2\pi f)t}$), and an equal negative frequency term ($e^{-j(2\pi f)t}$). We can ignore the $\frac{1}{2}$ because it is just a scaling term and we are more interested in the frequency analysis of our system. Since all the calculations that we are doing only involves linear operators (addition, subtraction, division, multiplication, derivative etc), we can apply the principle of superposition here. This means that we can analyze for the positive and negative frequencies separately, and simply sum the results later on.

Let us now assume that the voltage input to our capacitor is given by:

$$v(t) = Ae^{j(2\pi f)t}$$

$$\frac{dV(t)}{dt} = A(2\pi f)je^{j(2\pi f)t}$$

, where f is the frequency of the sinusoid that we are interested in. Take note that f is just a variable and it can represent both positive or negative frequencies, and A is simply a scaling coefficient.

We can then substitute this back into our capacitor model to arrive at the following equation:

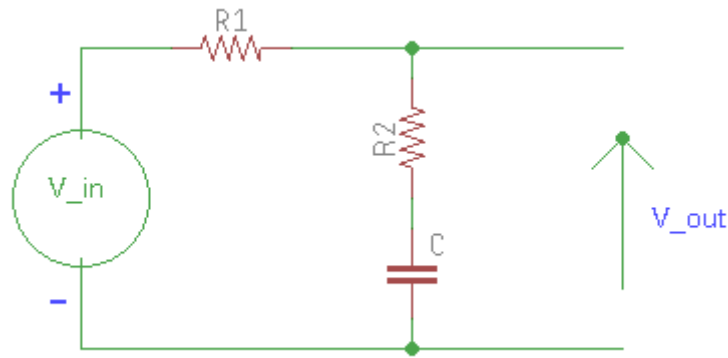
$$i(t) = (2\pi fC)j(Ae^{j(2\pi f)t})$$

$$i(t) = (2\pi fC)j(v(t))$$

$$\frac{v(t)}{i(t)} = \frac{1}{2\pi fCj}$$

From Ohm's law, we know that voltage divided by the current gives you the resistance. So from the above equation, the "resistance" of a capacitor is given by $\frac{1}{2\pi fCj}$. The official term for this is actually impedance because it is not really the same as the resistance. For one, this value is imaginary.

Now, we can start with a simple model as shown below.



SIMPLE CAPACITOR CIRCUIT

Capacitors in real life are not perfect. In order to model it better, we often use a parameter called the Equivalent Series Resistance (ESR) which you will usually see inside a data sheet for a capacitor. Let R_2 in the diagram be this ESR, and R_1 be the load that you are trying to operate. In order to analyze such a circuit, we just have to treat the capacitor as a normal resistor with a resistance (impedance actually) of $\frac{1}{2\pi f C j}$.

Applying the voltage divider rule and after some manipulation, we will arrive at the following equation:

$$\frac{V_{out}}{V_{in}} = \frac{1 + R_2(2\pi f C j)}{1 + (R_1 + R_2)(2\pi f C j)}$$

This is actually the Frequency Response of our circuit! You can substitute in the positive frequency ($V_{in} = A e^{j(2\pi f)t}$) to determine V_{out_1} for this positive frequency, and then do the same again for the negative frequency ($V_{in} = A e^{-j(2\pi f)t}$) to get another V_{out_2} for the negative frequency. Thereafter, you simply have to add (for cosine signals) or subtract (for sine signals) the 2 V_{out} to get the final answer. Seems tedious isn't it? But wait, there's actually an easier way out. Let me give you an example.

Suppose our input signal is given by:

$$V_{in} = 8 \cos(2t)$$

Since $2\pi f = 2$, the frequency of this input is $f = 1/\pi$. From Euler's Formula, we can decompose cosine into the sum of 2 exponential terms with positive and negative frequencies such as shown below.

$$8 \cos(2t) = 4e^{j2t} + 4e^{-j2t}$$

Next, let our Frequency Response be

$$\frac{V_{out}}{V_{in}} = \frac{3(2\pi f)j}{6 + 3(2\pi f)}$$

Substituting the 2 exponential terms from our input into the equation separately, we will get the following

$$V_{out_1} = \frac{3(2\pi\frac{1}{\pi})j}{6+3(2\pi\frac{1}{\pi})} * 4e^{j2t}$$

$$V_{out_2} = \frac{3(2\pi\frac{-1}{\pi})j}{6+3(2\pi\frac{-1}{\pi})} * 4e^{-j2t}$$

simplifying,

$$V_{out_1} = \frac{j}{1+j} * 4e^{j2t} \quad (1)$$

$$V_{out_2} = \frac{-j}{1-j} * 4e^{-j2t} \quad (2)$$

yes, i rigged the constants so that we get a simple equation. Anyway, we have to convert our imaginary fraction to exponential form now. The way to do it is to find its magnitude and argument (or phase). If you have forgotten about complex number manipulation, here are the rules:

Assuming $A = a + cj$ and $B = b + dj$,

$$|A| = \sqrt{a^2 + c^2}$$

$$|B| = \sqrt{b^2 + d^2}$$

$$|A * B| = |A| * |B|$$

$$\left| \frac{A}{B} \right| = \frac{|A|}{|B|}$$

$$\arg(A) = \tan^{-1}\left(\frac{c}{a}\right)$$

$$\arg(B) = \tan^{-1}\left(\frac{d}{b}\right)$$

$$\arg(A * B) = \arg(A) + \arg(B)$$

$$\arg\left(\frac{A}{B}\right) = \arg(A) - \arg(B)$$

So, back to our Frequency Response equation, we can now calculate the magnitude in the following manner:

$$\left| \frac{j}{1+j} \right| = \frac{1^2}{1^2+1^2} = \frac{1}{\sqrt{2}}$$

$$\left| \frac{-j}{1-j} \right| = \frac{(-1)^2}{1^2 + (-1)^2} = \frac{1}{\sqrt{2}}$$

Notice here that the magnitude is the same for the positive and negative frequencies. The argument (or phase) is then:

$$\arg\left(\frac{j}{1+j}\right) = \arg(j) - \arg(1+j) = 90^\circ - 45^\circ = 45^\circ$$

$$\arg\left(\frac{-j}{1-j}\right) = \arg(-j) - \arg(1-j) = -90^\circ - (-45^\circ) = -45^\circ$$

Therefore,

$$\frac{j}{1+j} = \frac{1}{\sqrt{2}} * e^{j45^\circ}$$

$$\frac{-j}{1-j} = \frac{1}{\sqrt{2}} * e^{j-45^\circ}$$

Now, we can further simplify our equation (1) and (2)

$$V_{out_1} = \left(\frac{1}{\sqrt{2}} * e^{j45^\circ}\right) * 4e^{j2t}$$

$$V_{out_2} = \left(\frac{1}{\sqrt{2}} * e^{j-45^\circ}\right) * 4e^{-j2t}$$

From the principles of superposition,

$$V_{out} = V_{out_1} + V_{out_2}$$

$$V_{out} = \frac{4e^{j(2t+45^\circ)} + 4e^{-j(2t+45^\circ)}}{\sqrt{2}} = 8 \frac{1}{\sqrt{2}} \cos(2t + 45^\circ)$$

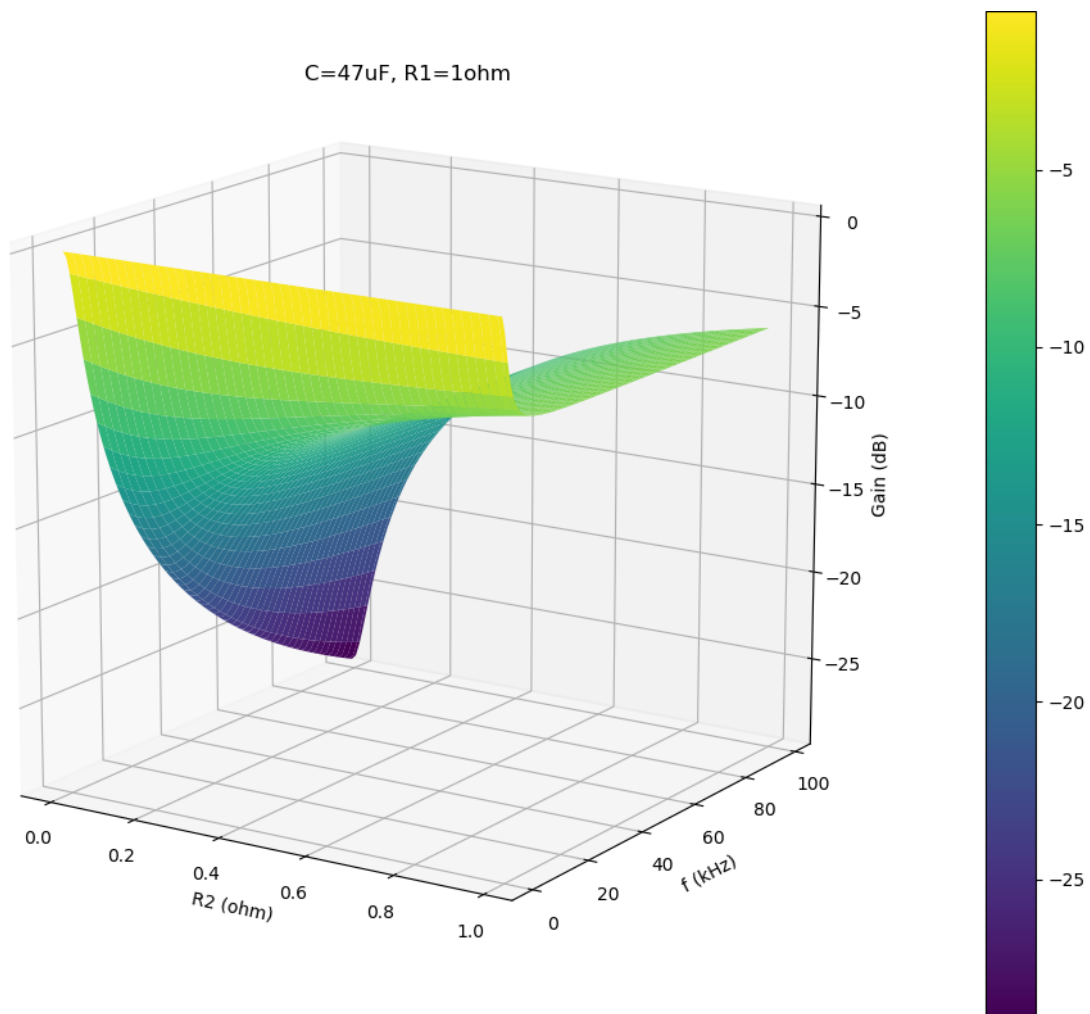
Finally we have the output for our input signal! Now, did you realize that if our signal is sine instead, our output will also be sine and the other constants will remain the same! Furthermore, our output amplitude is simply the input amplitude multiplied by the magnitude of the Frequency Response function (positive or negative), and our output phase is the input phase plus the phase of the Frequency Response function for our positive frequency. This means that it is sufficient to carry out analysis on only the positive exponential term! This is also the reason why signal related mathematics always use this kind ($v(t) = Ae^{j(2\pi f)t}$) of functions to analyze the system. I know this is a little bit of a hand-waving explanation but pardon me for this because I'm trying to put it as simply as possible.

Alright, for those who thought it was too long and didn't read, here's a summary. If you are interested in the amplitude of the resulting output V_{out} , you just have to take the amplitude of the input V_{in} and multiply it by the magnitude of the Frequency Response function, $\frac{1+R_2(2\pi f Cj)}{1+(R_1+R_2)(2\pi f Cj)}$. And if you are

interested in the resulting phase of the output, you can simply take the phase of your input and add it to the phase of the Frequency Response function. Try out this for a sine signal and convince yourself that it is true (:

We are not really interested in the phase for this analysis because we only want to know if our signal will be attenuated or amplified by the system. As such, we need to know the magnitude of the Frequency Response Equation. In order to determine the magnitude of a complex number, we have to square root the square of the real part plus the square of the imaginary part, namely $magnitude = \sqrt{real^2 + imaginary^2}$. But we will delegate this task to the almighty python because its just much simpler there.

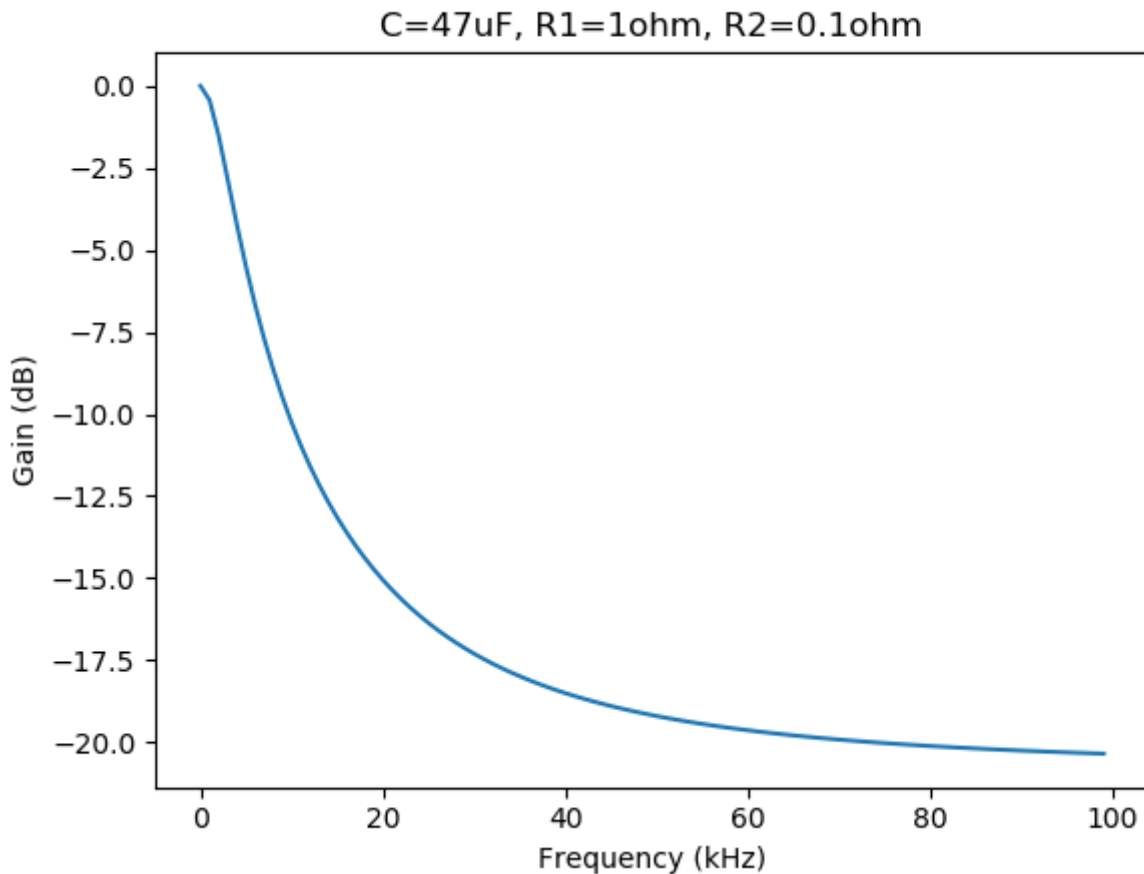
Now, let us put this equation into Python and plot it to see how the capacitance of our capacitor (C) affects the output voltage (V_{out}).



The values used to plot the graph above is $R1 = 1\text{ohm}$ and $C = 47\mu F$. The z-axis of the graph is in Logarithmic scale and it represents the gain of the system. To put it simply, it is the ratio of the amplitude of the output to the amplitude of the input. It is not really clear on the picture above but the values on the z-axis are actually negative so this means that the amplitude of the output is smaller than that of the input. For those of you who are not comfortable with the Log scale, the values on the scale is simply equal to $20 \log_{10} a$, where a is the value in normal scale. Thus a value of -20 on the log scale represents a ratio of $10^{-20/20} = 0.1$.

From the graph, you can tell that when the frequency increases, the gain decreases. This is especially more prominent when R_2 (the ESR of the capacitor) is low. The code to generate this graph is available below so do try it out to see how the different parameters affect the result.

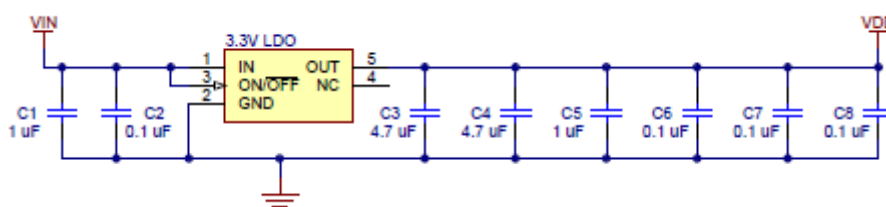
Now, let us take a cut out of the 3D graph to show the results for $R_2 = 0.1\text{ohm}$. Here's the resulting graph:



CAPACITOR FREQUENCY RESPONSE 3D

When the frequency is about 80kHz, the gain is approximately -20dB. In normal ratio, that is $10^{-20/20} = 0.1$. In other words, if you have frequencies above 80kHz in your signal, the amplitude of these high frequency signals will be approximately decreased by 90%, while lower frequencies will not be that much affected. This is why this type of circuit (where you connect a capacitor in parallel to the signal) is known as low pass filter. It only lets low frequencies pass.

This is also how you filter out high frequency noises in your system. You simply connect a capacitor to it and let it do the job. As an example, take a look at this schematic for this **voltage regulator** on a sensor board:



ACTUAL FILTER SCHEMATIC

Doesn't this look familiar now? The capacitors here are used to filter both the input voltage VIN and also the voltage output from the chip. When you put more than 1 capacitor to do the filtering such as that shown in the schematic, it is called a cascaded low pass filter. In general, the more capacitors you use, the better the performance of the filter in the sense that lower frequencies will not be attenuated much but higher frequencies will be attenuated more (slope of the 2D graph above will become more steep). Here's an [article](#) which talks about it more in detail for those who are interested.

Alright, that's it for the capacitor filter section! Great Job guys for making it till here! Just a little more before the end of this chapter!

Here's the Python script to plot the graphs shown above in case you want to try it out yourself. You can change the values of the parameters to see its effect on the circuit.

3D Plot

```

...
----R1-----
|           |           +
|           R2          |
Vin          |           Vout
|           C           |
|           |           -
-----
...

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
import math

PI = 3.141592653589793
numPoints = 100

R1 = 1
R2 = np.linspace(0, 1, numPoints, endpoint=False)
C = 47 * pow(10, -6)
f = np.linspace(0, 100, numPoints, endpoint=False)
X, Y = np.meshgrid(R2, f)
Z = np.zeros([numPoints, numPoints])

for i in range(0, numPoints):
    for j in range(0, numPoints):
        num = complex(1, R2[i] * 2 * PI * (f[j] * pow(10, 3)) * C)
        den = complex(1, (R1 + R2[i]) * 2 * PI * (f[j] * pow(10, 3)) * C)
        freqResponse = num / den
        magnitude = math.sqrt(pow(freqResponse.real, 2) + pow(freqResponse.imag, 2))
        Z[j][i] = 20 * math.log10(magnitude)

fig = plt.figure()
ax = Axes3D(fig)
cax = ax.plot_surface(X, Y, Z, cmap=cm.viridis)
ax.set_xlabel("R2 (ohm)")
ax.set_ylabel("f (kHz)")
ax.set_zlabel("Gain (dB)")
ax.set_title("C=47uF, R1=1ohm")
fig.colorbar(cax)

plt.show()

```

2D Plot

```

...
----R1-----
|           |           +
|           R2          +
Vin          |          Vout
|           C           -
|           |           -
-----
...

import numpy as np
import matplotlib.pyplot as plt
from matplotlib import cm
from mpl_toolkits.mplot3d import Axes3D
import math

PI = 3.141592653589793
numPoints = 100

R1 = 1
R2 = 0.1
C = 47 * pow(10, -6)
f = np.linspace(0, numPoints, numPoints, endpoint=False)
Z = np.zeros(numPoints)

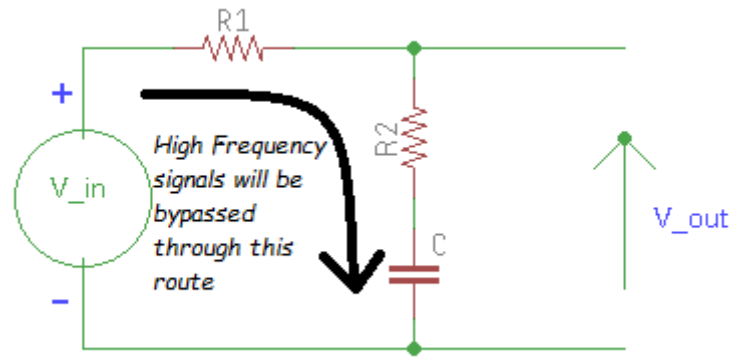
for j in range(0, numPoints):
    num = complex(1, R2 * 2 * PI * (f[j] * pow(10, 3)) * C)
    den = complex(1, (R1 + R2) * 2 * PI * (f[j] * pow(10, 3)) * C)
    freqResponse = num / den
    magnitude = math.sqrt(pow(freqResponse.real, 2) + pow(freqResponse.imag, 2))
    Z[j] = 20 * math.log10(magnitude)

plt.plot(f, Z)
plt.xlabel("Frequency (kHz)")
plt.ylabel("Gain (dB)")
plt.title("C=47uF, R1=1ohm, R2=0.1ohm")
plt.show()

```

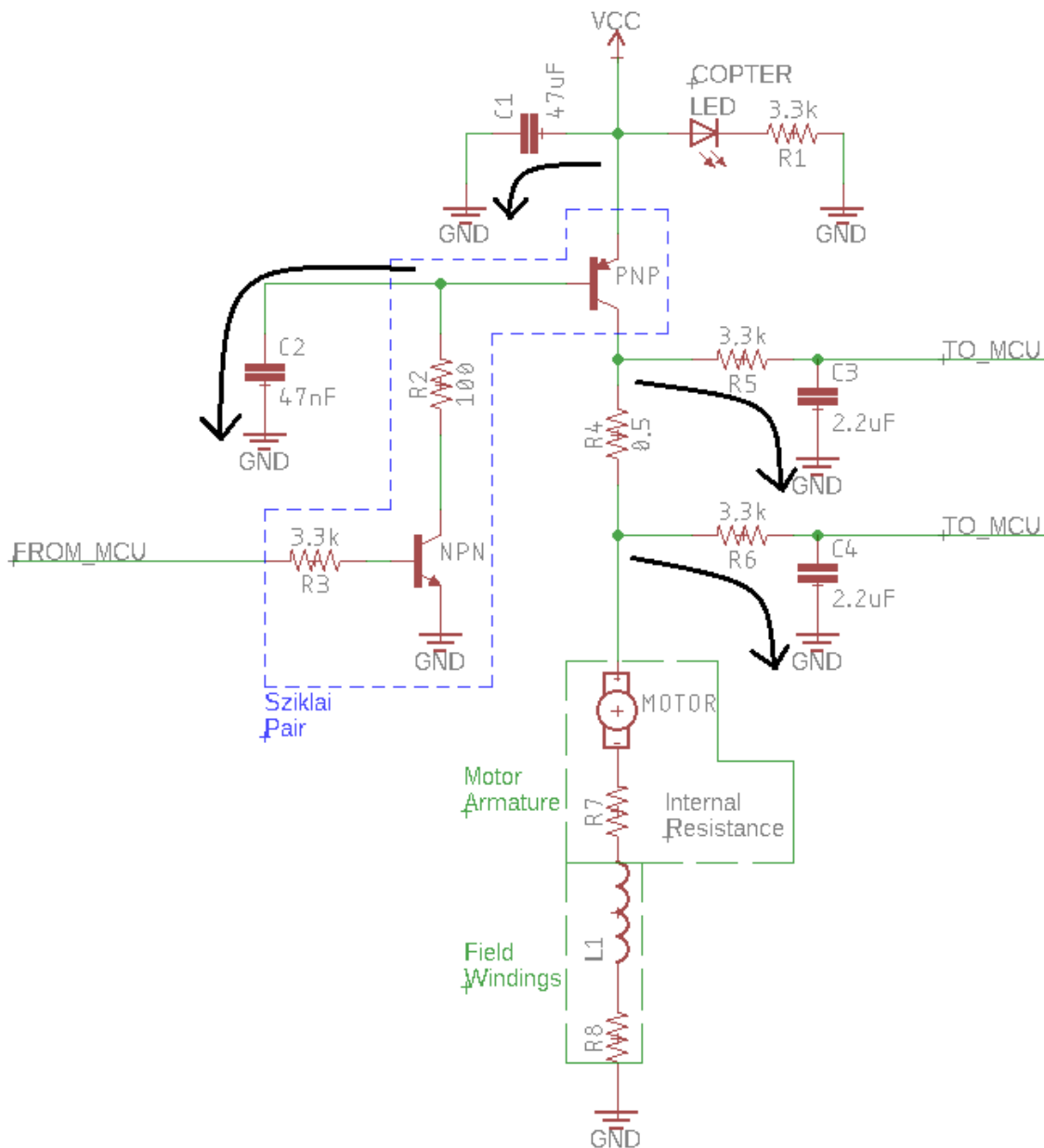
Putting Everything Together

From what we have learnt above, we know that the capacitor in our simple model acts as a low pass filter because it only lets low frequencies pass. Another name for such a capacitor is the bypass capacitor. This is because we can view it as high frequency signals being bypassed through the capacitor such as shown in the diagram below.



BYPASS CAPACITOR

Since high frequencies are bypassed, there is only low frequency signals in the output. Thus you will be able to get a stable voltage output in the above example (given that you choose the values of the capacitance and resistors appropriately). Now, let us use the same method to take a look again at our circuit.



CIRCUIT SCHEMATIC

I have marked out all the high frequency bypasses in the original diagram so lets go through them 1 by 1.

1. The first one at the top is to filter out high frequencies from our power source, VCC.
2. The second one on the left is to remove high frequencies from our switching circuit. This actually limits the frequency of our switching so if the Micro controller's signal is switching too fast, it will simply turn into a DC signal for the PNP transistor.
3. The 2 capacitors on the right side is to filter the signal which the Micro controller is going to read. This is important here because the Motor is very noisy (in the sense that it produce a lot of noise in our circuit due to the back EMF).

In our feedback control system, we will be required to take the derivative of our signals so it is important that there is as little noise as possible in our measurements. In order to determine the derivative of the signal, we have to take the difference between 2 points and divide it by the time difference (Δt).

$$\frac{dx}{dt} = \frac{x_1 - x_2}{\Delta t}$$

When the sampling rate is fast, Δt will be very small so the difference between 2 points will be divided by a small number, yielding a very huge value. Hence the noise will have a great impact on the result of our derivative.

Let us go now through the circuit that we are going to use. From our Arduino, we will give a signal to switch the NPN transistor on and off. Assuming that we are giving 5V to switch the NPN on, it will pull the Base of the PNP transistor to the ground, thereby turning the PNP transistor on (remember that PNP works in an opposite manner as compared to the NPN transistor). With the PNP transistor on, our power source, VCC, will then be able to reach our motor to turn the propeller.

On the other hand, when the Arduino gives a 0V to the NPN transistor, it will turn the NPN transistor off. Here's the tricky part. Under steady state, a capacitor acts like an open circuit. This is because in steady state, there will be no changes in states so any derivative term will be 0. As a result, $\frac{dV(t)}{dt} = 0$, and $i(t) = C \frac{dV(t)}{dt} = 0$. Current is 0A across the capacitor so this is akin to an open circuit where no current flows. Assuming that capacitor C2 goes into steady state (it will go to steady state as long as the switching is not faster than its voltage decay), the Base of the PNP transistor will be literally connected to nothing. Therefore, it will be "floating" and the PNP transistor will turn off. This cuts off the power supply for the motor so no torque will be provided to turn the propeller. The propeller thus slows down due to air resistance and friction.

Do you see how this circuit works now? By repeatedly switching on and off the circuit, we can actually control the speed on the propeller based on our switching speed. In this tutorial, our Arduino will provide a switching signal at a low current. The Sziklai Pair will then use our power source to "amplify" the current to turn our motor.

Alright folks, this marks the end of this chapter! I applaud you if you really read through everything in this article. If you have any questions, feel free to write it down in the comments section and I'll get back to you as soon as possible! Remember to take a break before moving on to the next chapter 😊

[Go to Next Section](#)

[PREVIOUS POST](#)[Hardware and Software Requirements](#)[NEXT POST](#)[Plotting Serial Data from Arduino in Real Time with Python](#)POSTED IN [FEEDBACK CONTROL SYSTEM](#)

4 comments on "*Building the Circuit and the Hardware*"

**Baris**

December 25, 2019 at 8:53 pm

Hello dear author,

I have a question for you .which program was used to draw Circuit schematic of the project?

Thanks a lot for everything

**Christian Klein**

December 9, 2020 at 6:17 am

Hey,

I saw you didn't use a flyback diode. I'm curious to know if this is on purpose.

Even though you more or less protected the analog inputs of your mcu, I can imagine the voltage spikes to be quite unhealthy for the pnp.

Regards,
Christian

**Roniel Velasco**

June 15, 2021 at 1:10 am

How did you connect the circuit to Arduino?


**Roniel Velasco**

June 15, 2021 at 1:10 am

How did you connect the circuit to Arduino?

Comments are closed.

Categories

Select Category 

Archives

- September 2019 (3)
- February 2019 (1)
- January 2019 (1)
- September 2018 (1)
- July 2018 (4)
- March 2018 (3)
- February 2018 (5)
- November 2017 (6)
- July 2017 (2)
- April 2017 (6)
- March 2017 (4)
- February 2017 (2)
- January 2017 (9)
- December 2016 (24)
- November 2016 (1)